CSIRO INEX Experiments: XML Search using PADRE

Anne-Marie Vercoustre, James A. Thom, Alexander Krumpholz, Ian Mathieson, Peter R. Wilkins, Mingfang Wu, Nick Craswell, David Hawking

CSIRO INEX experiments: XML search using PADRE

Anne-Marie Vercoustre[1] *James A. Thom*[2] *Alexander Krumpholz*[1] *Ian Mathieson*[1]

Peter Wilkins[1] *Mingfang Wu*[1] *Nick Craswell*[3]

David Hawking[3]

[1]CSIRO Mathematical and Information Sciences
Private Bag 10, South Clayton MDC, VIC 3169, Australia

[2]School of Computer Science and Information Technology, RMIT University
GPO Box 2476V, Melbourne 3001, Australia

[3]CSIRO Mathematical and Information Sciences
GPO Box 664, Canberra, ACT 2601, Australia

Email for correspondence: *Anne-Marie.Vercoustre@csiro.au*

Abstract

This paper reports on the CSIRO group's participation in INEX. We indexed documents and document fragments using PADRE the core of CSIRO's Panoptic Enterprise Search Engine. A query translator converts the INEX topics into queries containing selection and projection contraints for the results. Answers are extracted from ranked documents and document fragments based on the projection contraints in the query.

1 Introduction

Broadly speaking there are two main approaches to XML retrieval: a database approach as exemplified by query languages such as XQuery and a text retrieval approach as exemplified by search engines ranking documents or document fragments. The database and information retrieval communities have different approaches to query evaluation. The database community focuses on the expressive power of query languages that retrieve exact answers. The information retrieval community focuses on the effectiveness of ranked retrieval. Our approach at CSIRO to the INEX experiment was to add database techniques to an underlying text retrieval technology. Thus we combine selection and ranking of candidate documents and document fragments using information retrieval with a database style projection to extract the final answers. Further discussion of the motivation for our approach is described elsewhere [1].

We discuss issues with topic formulation in Section 2. In Section 3 we describe the overall architecture of our approach using PADRE, the core of CSIRO's Panoptic Enterprise Search Engine [2]. In Section 4 we outline the INEX runs we made and present our results.

2 Topics

Figure 1 shows topic 14, which is based on one of the topics proposed by our group to find figures that describe the Corba architecture and the paragraphs that refer to those figures. We are using this query in the rest of the paper as an example to describe our system.

As well as an obvious typographic error in the keywords, the topic finally used in INEX has several limitations. First, we did not correctly formulate the topic due to inadvertently overlooking some aspects of the complex DTD; there are other elements such as `<figw>` that should have logically been included in the topic. This raises a question for semi-structured retrieval — how much information about the structure is it reasonable to expect the average user to know? Second, due to the INEX requirement that answers could only be a single element it was not possible to capture the semantics as described in the narrative, that is an answer "would ideally contain both the figure and the paragraph referring to it". This could only happen in section elements which would have larger coverage than the specific information need. In defining the syntax and semantics for INEX topics it would have been desirable for different semantics to be given to

```
<te>fig,p</te>
```

meaning an answer would both be a `<fig>` element and a `<p>` element, whereas

```
<te>fig|p</te>
```

```
<?xml version="1.0"
    encoding="ISO-8859-1"?>
<!DOCTYPE INEX-Topic SYSTEM "inex-topics.dtd">
<INEX-Topic topic-id="14"
            query-type="CAS" ct-no="075">
<Title>
  <te>fig,p,ip1</te>
  <cw>Corba architecture</cw>
  <ce>fgc</ce>
  <cw>Figure Corba Architecture</cw>
  <ce>p, ip1</ce>
</Title>
<Description>
  Find figures that describe the Corba architecture
  and the paragraphs that refer to those figures.
</Description>
<Narrative>
  To be relevant a figure must describe the
  standard Corba architecture or a system
  architecture that relies heavily on Corba.
  A figure describing a particular aspect of a
  system will not be regarded as relevant even
  though the system may rely on Corba otherwise.
  Retrieved components would ideally contain both
  the figure and the paragraph referring to it.
</Narrative>
<Keywords>
  CORBA ORB Object Request Brocker Architecture
  interface invocation interoperability
  communication protocols IDL
</Keywords>
</INEX-Topic>
```

Figure 1: INEX topic 14

would mean an answer is either element. It is the former that the narrative of this topic implies.

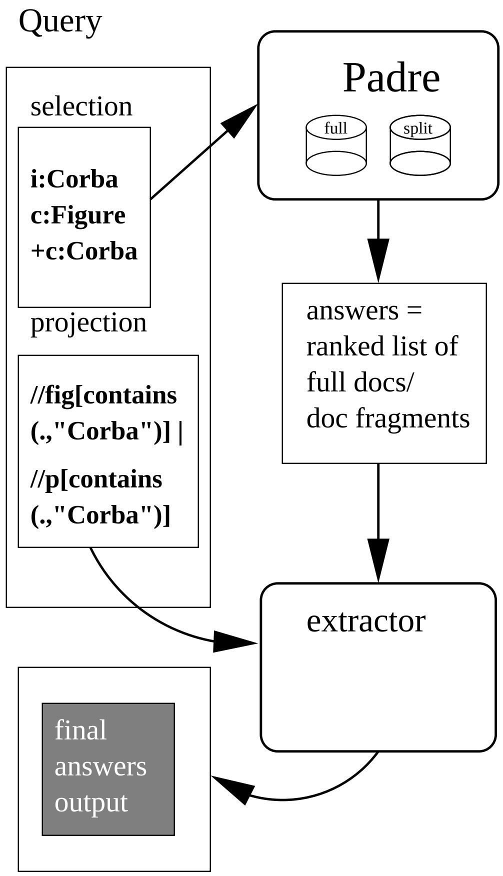
3 System overview

3.1 System Architecture

Figure 2 shows the overall architecture of our system. We translate INEX topics into queries comprising a selection component and a projection component; a simplified query is shown in the architecture diagram. The selection component of the query is sent to our search engine, PADRE, which ranks the more similar matching documents and document fragments meeting the selection criteria. The projection component, that is mostly based on the target element component of the topic, is sent to an extractor that extracts the desired answers from the ranked documents and document fragments returned by PADRE.

3.2 PADRE indexing

We extended CSIRO's document indexing and retrieval system, PADRE [3], to handle XML documents. PADRE is the indexing core of the Panoptic Enterprise Search Engine [2] and combines full-text



Query
selection
i:Corba
c:Figure
+c:Corba
projection
//fig[contains (.,"Corba")] |
//p[contains (.,"Corba")]
Padre
full
split
answers = ranked list of full docs/ doc fragments
extractor
final answers output

Figure 2: System architecture

and metadata indexing and retrieval. PADRE enables us to rank documents primarily on how many of the query terms appear in each document or document fragment and secondarily on the relevance score, using a slightly modified form of the Okapi BM25 function [4].

We were able to adapt PADRE's capability for indexing metadata fields to enable us to index selected XML elements. For example, given the mapping rule

```
//figc → i
```

the index terms for the element

```
<figc>Corba Architecture</figc>
```

would be mapped to the field "i" as `i:Corba` and `i:Architecture`.

As each element is processed, the first matching rule determines what metadata field is used to index the content of the element. In processing the content of sub-elements the rules are reapplied. Thus given the mapping rules

```
//p → c
//figc → i
```

`<figc><p>Corba Architecture</p></figc>` would be mapped to `c:Corba` and `c:Architecture`.

```
a - article author
b - bibliography entry
c - paragraph text (but not within abstract,
      keywords, acknowledgements etc)
d - publication date

f - figure text

i - figure caption
j - journal title
l - abstract, NOT including 's' keywords

n - acknowledgements

p - publisher
q - affiliation
r - table text
s - keywords
t - article title
u - url
v - fragment subset(s)
w - title of a section
y - ISSN
z - volume, issue, pp
```

Figure 3: Fields

This illustrates a weakness in our approach that higher structural elements are ignored.

The mappings are also used in queries. For example, the query “give me documents containing figures with Corba architecture in the caption” can be expressed as `i:Corba i:Architecture`. This query will first return matching documents that contain both “Corba” and “Architecture” in a figure caption, followed by partial matching documents that contain either “Corba” or “Architecture” in a figure caption. Mandatory constraints are supported, so this query could be expressed as `+i:Corba i:Architecture` so all matching documents must contain “Corba” in a figure caption. Phrase querying is also supported, in which case this query could be expressed as `i:"Corba Architecture"` and only documents containing the phrase “Corba Architecture” in the caption of a figure would be returned as answers.

A complete list of the fields is shown in Figure 3 together with the actual mappings in Figure 4

We only defined mappings for concepts that we considered useful for querying the INEX collection. The “v” field is used to allow queries on particular types of documents fragments.

3.3 Splitting

```
/books/journal/title → j
/books/journal/issue → z
/books/journal/publisher → p
/books/PANOPTIC-from → v
/books/PANOPTIC-genericXPath → v

/article/fm/hdr/hdr1/ti → j
/article/fm/hdr/hdr1/crt/issn → y
/article/fm/hdr/hdr2/obi → z
/article/fm/hdr/hdr2/pdt → d
/article/fm/hdr/hdr2/pp → z
/article/fm/tig/atl → t
/article/fm/tig/pn → z
/article/fm/au → a
/article/bdy/sec → c
/article/fm/abs → l
/article/fm/abs/p → l
/article/PANOPTIC-from → v
/article/PANOPTIC-genericXPath → v

//ack → n
//ack/p → n
//kwd → s
//kwd/p → s
//aff → q
//url → u
//st → w
//bb → b

//p → c
//p1 → c
//p2 → c
//p3 → c
//ip1 → c
//ip2 → c
//ip3 → c
//ip4 → c
//ip5 → c
//ilrj → c
//item-none → c
//fig → f
//figw → f
//fgc → i
//tbl → r
```

Figure 4: Actual mappings

As shown in Figure 2 the system uses PADRE to select and rank documents. We wanted to make good use of PADRE’s initial ranking and, since Wilkinson [5] shows that simply extracting elements from ranked documents is a poor strategy, we decided to investigate ranking document fragments as well as whole documents. Thus before indexing by PADRE we split the documents into various fragments and indexed the fragments as well as the whole documents. For the content only queries we expected that ranking document fragments as well as whole documents will improve performance by finding the relevant portions of documents, especially where the coverage of whole documents was too broad. For the content and structure queries we expected the splitting to improve the ranking but also envisaged that for queries involving a specific target element

```
/article/
/article/bdy//fig/
/article/bdy//figw/
/article/bdy//ilrj/
/article/bdy//ip1/
/article/bdy//ip2/
/article/bdy//ip3/
/article/bdy//ip4/
/article/bdy//ip5/
/article/bdy//item-none/
/article/bdy//p/
/article/bdy//p1/
/article/bdy//p2/
/article/bdy//p3/
/article/bdy/sec/
/article/bdy/tbl/
/article/fm/
/article/fm/abs/
/books/
```

Figure 5: Document fragments

further extracting would be required. We describe this further in the next section.

We analysed the collection and identified elements to use as fragments based on:

- a reasonable granularity that is not too small, and
- the expected elements for results.

Thus we split document fragments based on the paths shown in Figure 5 We also included some additional context to the fragments such as the filename of the original document and the path within the document to the fragment. This context allows subsequent processing of the document fragment.

We were able to use our existing indexing and retrieval engine to index both the documents and the fragments as one collection although this increased the number of "documents" by a factor of 100, and the size in bytes by a factor of 10.

If the query does not contain a projection, then the result of query is simply the ranked list produced by PADRE. Otherwise the extractor described in the next section is applied to the ranked list of documents and document fragments.

3.4 Extractor

Many of the content and structure queries contain a projection. We automatically generate the projection when there is a target element in the topic. Example of a projection in a query corresponding to topic 14 is shown in Figure 6. The projection is an XPath specifying the target element or elements to be extracted from the ranked list of documents and document fragments. The algorithm is as follows, for each returned fragment f:

```
1. load the fragment, get the name of the embedding article, load the full article $A$.

2. apply the XPath projection to the article $A$; this returns $e_1, e_2, \dots e_n$ elements.

3. $g = f$

4. while $g != nil$ do
   if ($g == e_i$ for any $e_i$)
   then return the XPath of $g$ and exit
   else calculate $g = parent(g)$

5. if (there are $e_i$ that are descendants of $f$)
   then return all of those and exit
   else return the $e_i$ (if any)
```

```
</query>
<query topic-id="14">
<selection>
  i:Corba
  i:architecture
  c:Figure
  c:Corba
  c:Architecture
  [CORBA ORB Object Request Brocker
   Architecture interface invocation
   interoperability communication
   protocols IDL]
</selection>
<projection>
  //fig |
  //p[contains(.,"Figure") or
      contains(.,"figure") or
      contains(.,"Corba") or
      contains(.,"corba") or
      contains(.,"Architecture") or
      contains(.,"architecture")] |
  //ip1[contains(.,"Figure") or
        contains(.,"figure") or
        contains(.,"Corba") or
        contains(.,"corba") or
        contains(.,"Architecture") or
        contains(.,"architecture")]
</projection>
</query>
```

Figure 6: Query for topic 14

After our inital submission, we looked at improving the order of our final answers. We identified key terms in the projection, in the example of topic 14 "Corba", "Figure", and "Architecture". By globally ranking the extracted fragments into tiers based on how many of the key terms appear in the projected elements, irrespective of how many times they appear and ignoring upper and lowercase differences.

3.5 Query Translator

The query translator constructed queries that we could process with our search engine and extractor.

Figure 6 shows the query that was automatically generated for topic 14.

The following process was developed by analysing the structure of the topics in order to deduce the semantics of the various possible constructs in a topic, particularly the `<Title>` of a topic.

The `<cw>` and `<ce>` elements in the title of the topic are used to generate the selection component of the query. Mappings, similar to those described in Section 3.2 for the indexing, are used to map paths within `<ce>` elements to PADRE fields.

If there is more than one field specified by the paths within a `<ce>` element, then all possible combinations of the field mappings from the `<ce>` with terms from the `<cw>` must be generated in the query.

When content element involves dates, we use the metadata field “`d`” and convert the `<cw>` element into constraints on numerical dates. Similarly we attempt to identify phrases using location of commas in topic, so as to take advantage of the phrase feature of PADRE.

The `<te>` target element if present is translated into the projection component of the query. When the path in the projection maps to a field also used in the selection component additional contraints should be added to the projection.

4 Experiments and Results

We submitted three official runs to INEX:

- queries on *full* articles (run 1)
- queries on *split* articles (run 2)
- *manually* contructed queries on split articles (run 3)

Subsequently we also explored:

- queries on split articles with *post-projection fragment reranking* (run 4)

and corrected a bug with run 1:

- queries on *full* articles – revised (run 5)

Results for runs 2, 3, and 5 on both the content-and-structure (CAS) and content-only (CO) topics are shown in Figures 8, 9, and 10 respectively. These figures show results for our runs (wide red line) with a comparison to other systems.

We also analysed results on topic 14 in more depth. Results for runs 2, 3, 4 and 5 on topic 14 are shown in Figures 11, 12, 13 and 14 respectively. These graphs show relevance judgements for the 100 highest ranked answers for each run. Each answer corresponds to a vertical bar of about 2mm width. The highest ranked answer appears on the left. The height of the vertical bars represents the degree of relevance, and the greylevel the coverage. For comparison we have also included the optimal ranking in Figure 15 which shows there is still considerable room for further improvement in XML retrieval.

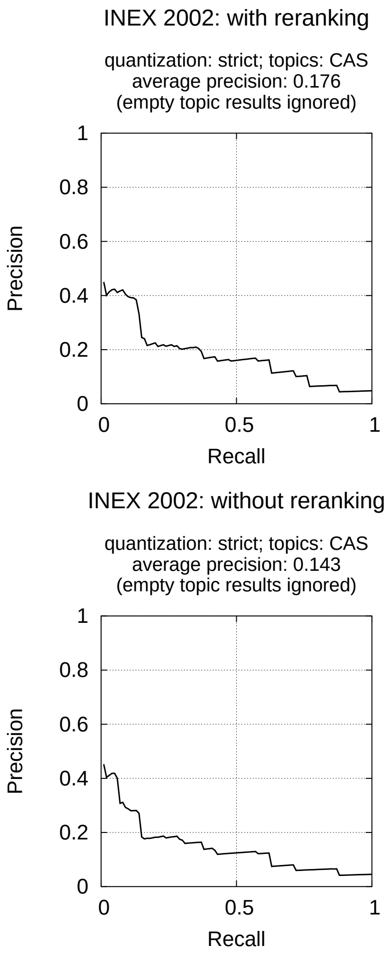


Figure 7: Nine queries on split articles (run 4) with and without post-projection reranking

Results for run 4 on a limited set of topics is shown in Figure 7. The reranking could only be applied to nine queries where the target elements also appear within the content word constraints. For such queries the post-projection reranking of fragments is effective as many unjudged elements were returned. This is very clearly borne out with topic 14, as can be seen from comparing Figure 14 with Figure 12. Overall the performance of the nine queries with reranking (top graph in Figure 7) is better than without reranking (bottom graph in Figure 7).

In topic 14 the manually constructed query performed worse than the automatically generated query using the query translator. However as shown in Figures 9 and 10 generally the manually constructed queries performed much better than the automatically generated queries for the CAS topics. But this was not the case for the CO topics as shown in Figures 9 and 10, perhaps because less effort was spent on improving these queries.

Our draft version of this paper presented at the INEX workshop as well as another of our papers [1] has a claim, based on the erroneous run 1, that using the collection containing documents and document fragments (run 2) was more effective than using just the full documents. However the new run for the full documents (run 5) invalidates this claim as shown by comparing Figure 10 and Figure 8, in fact the split performed worse.

A key question that the INEX experiments has not addressed is do users want to get back documents fragments or are they more interested in pointers to relevant parts within actual documents. This raises questions about what constitutes an answer and how answers should be organised when presented to the user.

References

[1] N. Craswell, D. Hawking, A. Krumpholz, I. Mathieson, J. A. Thom, A.-M. Vercoustre, P. Wilkins and M. Wu. XML document retrieval with PADRE. In *Proceedings of the 7th Australasian Document Computing Symposium*, Sydney, Australia, 16 December 2002.

[2] CSIRO and Australian National University. Panoptic enterprise search engine. `http://www.panopticsearch.com/`.

[3] David Hawking, Peter Bailey and Nick Craswell. Efficient and flexible search using text and metadata. Technical Report TR2000-83, CSIRO Mathematical and Information Sciences, 2000. `http://www.ted.cmis.csiro.au/~dave/TR2000-83.ps.gz`.

[4] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu and M. Gatford. Okapi at TREC-3. In D. K. Harman (editor), *Proceedings of TREC-3*, Gaithersburg MD, November 1994. NIST special publication 500-225. `http://trec.nist.gov/pubs/trec3/papers/city.ps.gz`.

[5] R. Wilkinson. Effective retrieval of structured documents. In W. B. Croft and C.J. van Rijsbergen (editors), *Proceedings of the 17th Annual International Conference on Research and Development in Information Retrieval*, pages 311–317, Dublin, Ireland, July 3-6 1994. Springer-Verlag.

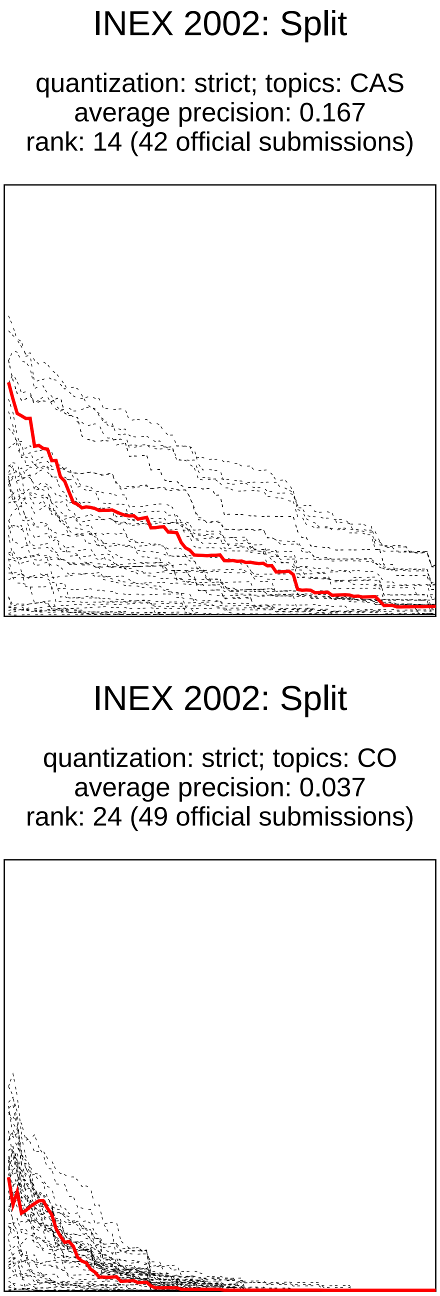
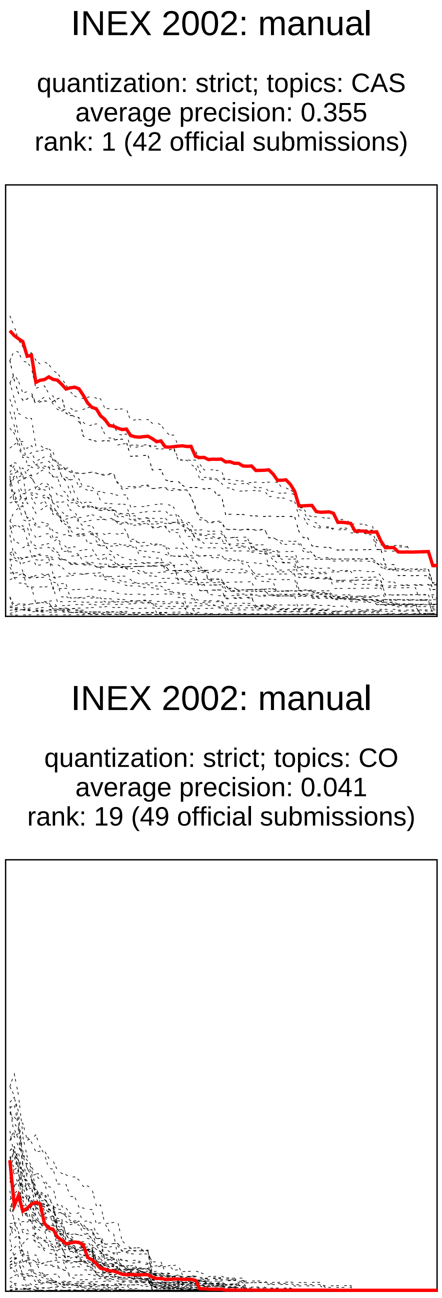


Figure 8: Queries on split articles (run 2)



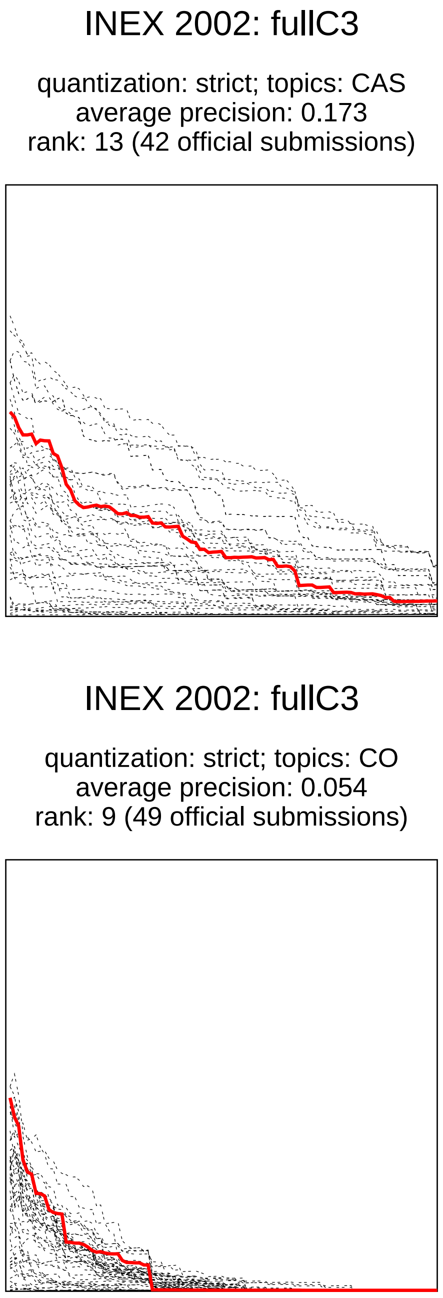
INEX 2002: manual

quantization: strict; topics: CAS
average precision: 0.355
rank: 1 (42 official submissions)

INEX 2002: manual

quantization: strict; topics: CO
average precision: 0.041
rank: 19 (49 official submissions)

Figure 9: Manually improved queries (run 3)



INEX 2002: fullC3

quantization: strict; topics: CAS
average precision: 0.173
rank: 13 (42 official submissions)

INEX 2002: fullC3

quantization: strict; topics: CO
average precision: 0.054
rank: 9 (49 official submissions)

Figure 10: Queries on full articles (run 5)

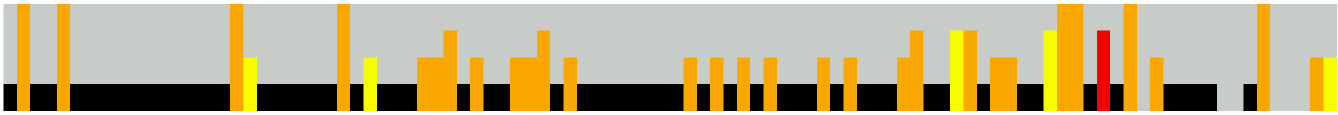


Figure 11: Results for Topic 14 — query on full articles (run 5)

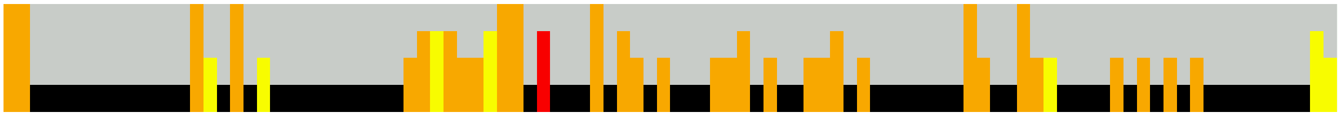


Figure 12: Results for Topic 14 — query on split articles (run 2)

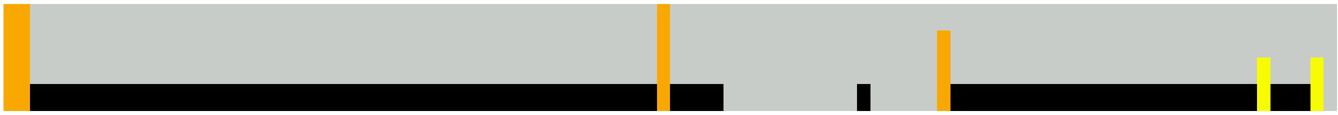


Figure 13: Results for Topic 14 — manual queries (run 3)

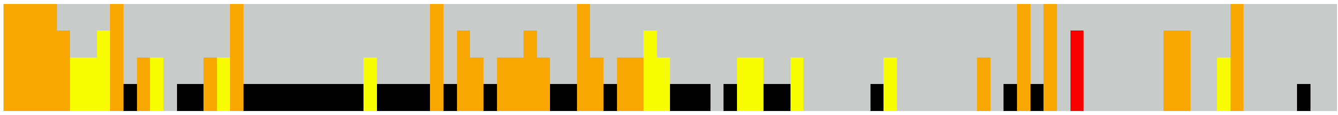


Figure 14: Results for Topic 14 - query on split articles with further reranking of final answers (run 4)

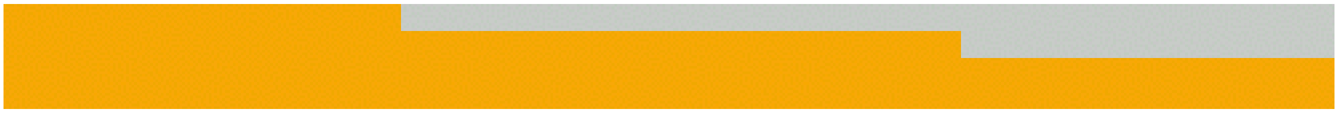
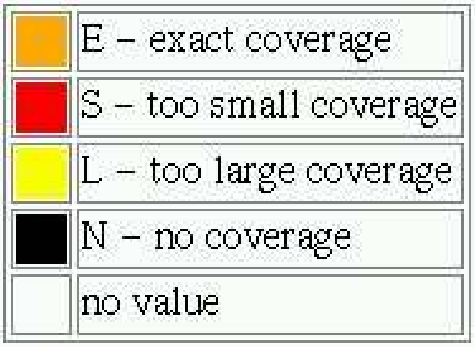


Figure 15: Results for Topic 14 - optimal ranking (relevance only) with first version of relevance judgements

In the above figures, the results are shown from left (highest ranked) to right. The height of the bar represents the relevance and the colour of the bar indicates the coverage as shown below:



🟧	E – exact coverage
🟥	S – too small coverage
🟨	L – too large coverage
⬛	N – no coverage
⬜	no value